In re the application of

Hannigan et al.

Application No.: 09/870,563

Filed: 05/31/2001

Date: March 30, 2005

Group Art Unit: 2126

Examiner: Zhen, Li B

From: Tucson, AZ 85744

Amendments to the Claims

Listing of Claims

This listing of the claims will replace all prior versions, and listings, of claims in the application.

CLAIMS

1. (Currently amended) A method for bridging messages between a first and at least a second application having differing message formats, said method comprising:

dynamically loading a message bridge module into said first application;

operating said message bridge module by employing a userexit routine;

setting a pointer to said userexit routine by making an operating system call to load a userexit module into said first application's memory and workspace;

resolving said userexit routine at the execution of said first application, not at the compilation of said first application;

said first application outputting data to said userexit module without receiving any confirmation regarding whether said data was received;

said bridge module receiving message data from ~~the adapter~~ said userexit module of said first application in a first format;

said bridge module translating and/or parsing said received message data into at least a second format; and

said bridge module outputting said translated and/or parsed message data to at least said second application.

2. (Cancelled)

3. (Currently amended) A method as claimed in claim 1, ~~further~~ wherein said message data is event data.

4. (Cancelled)

5. (Cancelled)

6. (Cancelled)

7. (Original) A method as claimed in claim 1, wherein said translating and/or parsing is performed by instructions residing on the same server as said first application.

8. (Original) A method as claimed in claim 1, further comprising handling errors for messages outputted to at least said second application.

9. (Original) A method as claimed in claim 1, further comprising queuing messages outputted to at least said second application.

10. (Original) A method as claimed in claim 1, further comprising determining the message type of the message data received.

11. (Currently amended) A computer system comprising:

a first application having an adapter capable of outputting message data in a first format;

at least a second application capable of receiving message data in a second format;

a message bridge module dynamically loaded into said first application, said message bridge module operating by employing a userexit routine and adapted to receive message data from the adapter of said first application in said first format;

said first application adapted for;

setting a pointer to said userexit routine by making an operating system call to load a userexit module into said first application's memory and workspace;

resolving said userexit routine at the execution of said first application, not at the compilation of said first application;

outputting data to said userexit module without receiving any confirmation regarding whether said data was received;

said message bridge module further adapted to translate and/or parse said received message data into at least said second format[[;]] and output said translated and/or parsed message data to at least said second application.

12. (Currently amended) A computer system as claimed in claim 11, wherein said message bridge module resides on the same server as said first application.

13. (Currently amended) A computer system as claimed in claim 11, wherein said message data is event data ~~herein said adapter is a generic adapter or a userexit~~.

14. (Cancelled)

15. (Cancelled)

16. (Cancelled)

17. (Currently amended) A computer system as claimed in claim 11, further comprising an error handler for messages outputted to at least said second application.

18. (Currently amended) A computer system as claimed in claim 11, further comprising a message queue for messages outputted to at least said second application.

19. (Currently amended) A software module, stored on a computer-readable medium, for bridging messages between a first and at least a second application having differing message formats, said module comprising:

instructions for dynamically loading a message bridge module into said first application;

instructions for operating said message bridge module by employing a userexit routine;

instructions for setting a pointer to said userexit routine by making an operating system call to load a userexit module into said first application's memory and workspace;

instructions for resolving said userexit routine at the execution of said first application, not at the compilation of said first application;

instructions for said first application outputting data to said userexit module without receiving any confirmation regarding whether said data was received;

instructions for said bridge module receiving message data from ~~the adapter~~ said userexit module of said first application in a first format;

instructions for said bridge module translating and/or parsing said received message data into at least a second format; and

instructions for said bridge module outputting said translated and/or parsed message data to at least said second application.

~~instructions for receiving message data from the adapter of said first application in a first format;~~

~~instructions for translating and/or parsing said received message data into at least a second format; and~~

~~instructions for outputting said translated and/or parsed message data to at least said second application.~~

20. (Cancelled)

21. (Original) A software module as claimed in claim 19, wherein said message data is event data.

22. (Cancelled)

23. (Cancelled)

24. (Cancelled)

25. (Cancelled)

26. (Original) A software module as claimed in claim 19, further comprising instructions for handling errors for messages outputted to at least said second application.

27. (Original) A software module as claimed in claim 19, further comprising instructions for queuing messages outputted to at least said second application.

28. (Original) A software module as claimed in claim 19, further comprising instructions for determining the message type of the message data received.

29. (Currently amended) A message bridging apparatus comprising:

storage means for storing computer-readable instructions;

instructions, stored in said storage means for dynamically loading a message bridge module into said first application;

instructions, stored in said storage means for operating said message bridge module by employing a userexit routine;

instructions, stored in said storage means for setting a pointer to said userexit routine by making an operating system call to load a userexit module into said first application's memory and workspace;

instructions, stored in said storage means for resolving said userexit routine at the execution of said first application, not at the compilation of said first application;

instructions, stored in said storage means for said first application outputting data to said userexit module without receiving any confirmation regarding whether said data was received;

instructions, stored in said storage means for said bridge module receiving message data from said userexit module of said first application in a first format;

instructions, stored in said storage means for said bridge module translating and/or parsing said received message data into at least a second format; and

instructions, stored in said storage means for said bridge module outputting said translated and/or parsed message data to at least said second application.

~~instructions, stored in said storage means, for receiving message data from a first application in a first format, said first application having an adapter;~~

~~instructions, stored in said storage means, for translating and/or parsing said received message data into at least a second format; and~~

~~instructions, stored in said storage means, for outputting said translated and/or parsed message data to at least a second application.~~

30. (Cancelled)

31. (Original) A message bridging apparatus as claimed in claim 29, wherein said message data is event data.

32. (Cancelled)

33. (Cancelled)

34. (Cancelled)

35. (Cancelled)

36. (Original) A message bridging apparatus as claimed in claim 29, further comprising instructions for handling errors for messages outputted to at least said second application.

37. (Original) A message bridging apparatus as claimed in claim 29, further comprising instructions for queuing messages outputted to at least said second application.

38. (Original) A message bridging apparatus as claimed in claim 29, further comprising instructions for determining the message type of the message data received.